Lecture 11: Views in SQL

- More Database Commands
  - DDL, DML, DCL

- Views
  - Logical Independence
  - View Definitions
  - When to Use Views
  - Updating Views

Database Language Commands

- DDL: Data definition language – to create and modify the schema
- DML: Data manipulation language – to create and modify the data (the rows)
- DCL: Data control language – to handle security (grant and revoke privileges); may not be covered in this class

DDL: Data Definition Language

- Create, edit, or delete database objects
  - Tables
  - Stored Procedures
  - Data Types
  - NOT ROWS!
- Drop table:

DROP TABLE Patient;

DDL: CREATE TABLE (SQL SERVER)

```
CREATE TABLE Patient (
    id uniqueidentifier NOT NULL,
    FirstName varchar(50),
    MiddleName varchar(30),
    LastName varchar(50),
    Provider uniqueidentifier,
    DateOfBirth datetime,
    CONSTRAINT pk_Patient PRIMARY KEY (id)
    CONSTRAINT fk_Patient FOREIGN KEY (Provider)
    REFERENCES (Staff.id)
)
```

DDL: ALTER TABLE

ALTER TABLE Patient
- [ADD COLUMN]
- [ALTER COLUMN]
- [DROP COLUMN]
- [ADD CONSTRAINT]
- [DROP CONSTRAINT]

DML: Data Manipulation Language

- Inserting, updating, or deleting rows

- Delete:

```
DELETE FROM Patient
WHERE FirstName LIKE 'B%'
```

INSERT INTO

INSERT INTO Patient (id, FirstName, LastName)
VALUES (4, ‘Bob’, ‘Thomas’), (6, ‘Ned’, ‘Flanders’)

INSERT INTO Patient (id, FirstName, DateOfBirth)
SELECT Guid, Fname, DOB FROM Staff WHERE
IsPatient = 1

UPDATE

```
UPDATE TABLE Patient
SET  FirstName = 'Bob',
       DateOfBirth = AddDays(DateOfBirth, 1)
WHERE Provider = 8
```

Views

- A *view* is a query, with a name, that is stored in the database.

- Example view definition:

```
CREATE VIEW gstudents AS
   SELECT S.*
   FROM student S  WHERE s.gpa >= 2.5
```

- Views can be used like *base tables*, in a query or in any other view. Views are like macros.

Views can be used to make queries simpler

Completed(StudID, Course)

```
CREATE VIEW gstudents AS
   SELECT S.*
   FROM student S  WHERE s.gpa >= 2.5
```

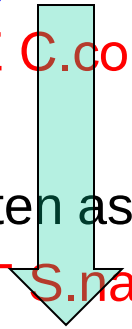
- Queries about “good” students only are easier to write

```
SELECT S.name, S.phone
FROM gstudent S NATURAL JOIN completed C
WHERE C.course = ‘CS386’;
```

How are views implemented?

When you enter a query that mentions a view in the from clause, the DBMS expands/rewrites your query to include the view definition.

```
SELECT S.name, S.phone
FROM gstudent S NATURAL JOIN completed C
WHERE C.course = 'CS386';
```



is rewritten as:

```
SELECT S.name, S.phone
FROM (SELECT S.* FROM student S  WHERE s.gpa >= 2.5) AS S
    NATURAL JOIN completed C
WHERE C.course = 'CS386';
```

Views used for Security

- This views presents a table (called sstudent) that is the student relation without the gpa field.

```
CREATE VIEW sstudent AS
   SELECT studID, name, address
   FROM student
```

- Can you think of some other security examples?

Views used to support extensibility

- A company's database includes a relation:

  Part (PartID: Char(4), weight: real,…)

- Weight is stored in pounds
- Company is purchased by a firm that uses metric weights
- Databases must be integrated and the weight must be in metric units
- But there's much old software using pounds.
- Solution: views!

Views to support extensibility (cont.)

Solution:

1. Define a new base table with kilograms, NewPart, for the integrated company.

2. CREATE VIEW Part AS
   SELECT PartID, 2.2046*weight, …(no other changes)…
   FROM NewPart

3. Old programs still call the table “Part”; now they are referencing a view; before they references the table.

What if you want to update the view? (It looks like a table …)

- Views cannot always be updated unambiguously
- Consider Students(StudID, gpa, major,…)

```
CREATE VIEW majorgpa AS
    SELECT major, AVG(gpa)
    FROM Students
    GROUP BY major
```

Majorgpa

major	gpa
CS	3.5
ECE	3.5

Updatability of views

- I want to change the GPA of CS majors from 3.5 to 3.6 .
- How can I do that? (Should I do that?)

Restrictions for Updateable views

- A view can be updated if
  - It is defined on a single base table
  - No aggregates, group by
  - No DISTINCT
  - No set operations

- Notes:
  - Products differ in how they define updateable view
  - On the research front, there are more cases where a view can be updateable

Restrictions for Insertable views

To insert into a view, the view must:

- Not have use a column from the base table multiple times
- Contain all columns that do not have a default value; therefore, it must include the key(s) for the base table
- Not have any derived columns (using arithmetic or constants)

Example 1

student(id, name, gpa, major)

```
CREATE VIEW gstudents AS
   SELECT S.*
   FROM student S  WHERE s.gpa >= 2.5
```

```
UPDATE gstudents s
   SET major = 'CSE'
   where gpa > 3.5
```

This is allowed; all rows where gpa > 3.5 will have their major changed

Example 2: (key is NOT in the view)

student(id, name, gpa, major)

```
CREATE VIEW gstudents AS
   SELECT S.name, s.gpa, s.major
   FROM student S  WHERE s.gpa >= 2.5
INSERT INTO gstudents s
   VALUES ("Smith, Bob", 3.72, "CS")
```

This is NOT allowed because we can't insert a row into student (the underlying table) without providing a value for id (the key).

Data Independence

- A database management system supports *data independence* if application programs are immune to changes in the conceptual and physical schemas.
- Why is this important? Because everything can change.
- How does the relational model achieve logical (conceptual) data independence?

Levels of Abstraction – support Data Independence

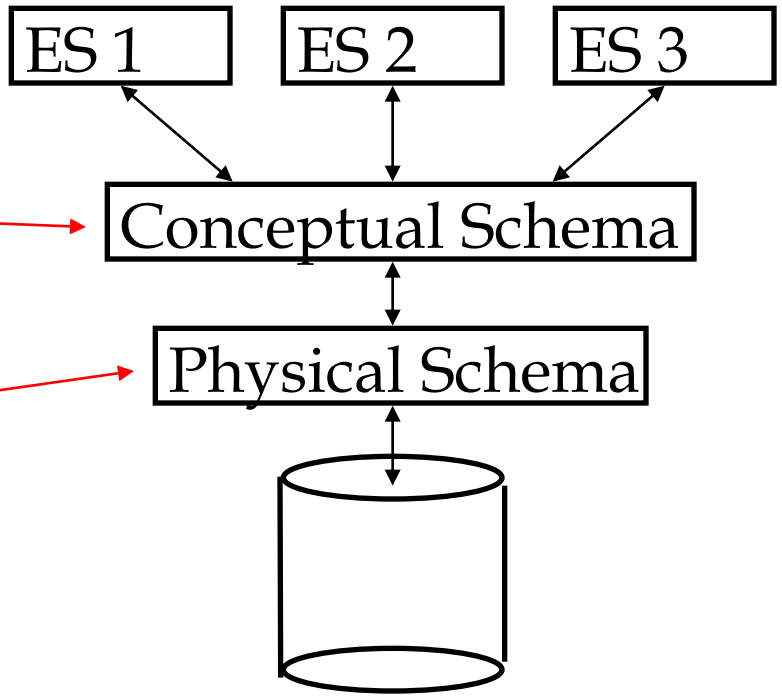
External view; user and data designer

Logical data indepence

Logical storage; data designer

Physical data indepence

Physical storage; DBA

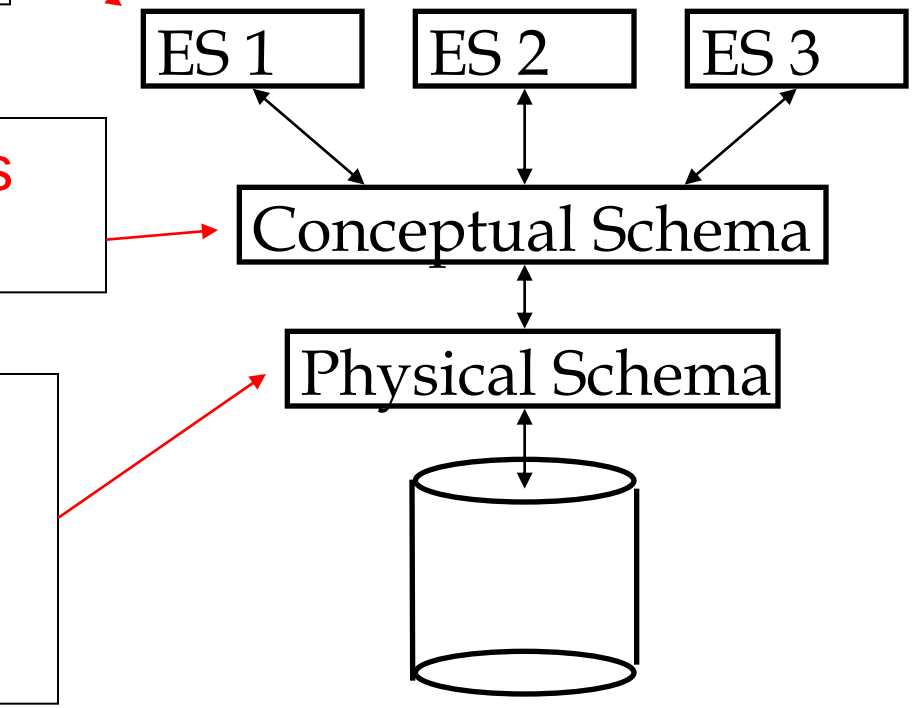


Levels of Abstraction in a relational DBMS

External schemas are Views

Conceptual schema: ERDs
Logical schema: tables

Physical schema: the tables – after they are modified by the DBA, plus indexes, files, etc.



A few Comments on Views

- View versus temporary table
  - What is the difference?
  - Which is better?

- Some DBMS' offer a way around the view update problem
  - Triggers
  - Rules
  - Active research

Summary

- A view is a stored query definition; rows are not computed until they are needed.
- Views can be very useful
  - Easier query writing, security, extensibility
- Views cannot always be unambiguously updated
- Three levels of abstraction in a DBMS supports data independence: logical and physical

Next slide is relevant to Lecture 12 (delivered by Eric Wheeler) on embedded SQL

Rowsets

- Different from a RecordSet!
- Rowset objects are available in both Java and .Net implementations
  - .Net calls them “DataTables”
- In-memory abstraction of a table
- Allows for in-place editing
  - Don’t forget to commit!
- A Rowset is an interface
  - Each DBMS vendor must provide an implementation – why?